# Mini Project Report *on*

**“Teacher Management System”**

### Submitted by,

|  |  |  |
| --- | --- | --- |
| Adarsh Malviya | PF 31 | 1032180763 |
| Tanvi Poddar | PF 24 | 1032180423 |
| Dhruv Samadhiya | PF 28 | 1032180595 |
| Khuzema Khomosi | PF 59 | 1032181584 |

**Under Guidance of**

**Prof. Shilpa Sonawani**

At



## School of Computer Engineering and Technology

**Contents**

[Abstract I](#_TOC_250003)

[List of Figures II](#_TOC_250002)

[List of Tables III](#_TOC_250001)

[1. Introduction 1](#_TOC_250000)

a. Motivation ……………………………………………………………...

b. Objectives ………………………………………………………….…...

### 2. Problem Statement …………………………………………………………….2

a. Scope …………………………………………………………………...

b. Planning ………………………………………………………………..

### Tools & Technologies Used ……………………………………………………3

* 1. Software Requirements ………………………………………………...
  2. Hardware Requirements ………………………………………………..

### 4. Database Design ………………………………………………………………...4

1. Functional Requirements ……………………………………………..…
2. Non-Functional Requirements ………………………………………….

c. ER Diagram …………………………………………………………….

### 5. Database Schema ……………………………………………………………….5

### 6. DDL……………………………………………………………….6

### 7.DML……………………………………………………………….7

### 8.DCL……………………………………………………………….8

### 9.TRIGGERS……………………………………………………………….9

### 10.PLSQL Procedure/Function……………………………………………………………….10

### 11.Frontend GUI ……………………………………………………………….11

1. **Relational Database Design ………………………………………………...……12**
2. **Normalized Database Design ……………………………………………………13**

Conclusion …………………………………………………………………..…...14

## Abstract

An organized and systematic office solution is essential for all universities and organizations. We are using a web-based teacher management system, which can be implemented on any computer. This system is being developed to maintain easy access of information from the database. The application makes use of React.JS Framework for front-end and My-Sql as backend. It tracks all the details of a teacher's attendance from day one to end of the course.

Teacher performance management is a continuous process for identifying, evaluating and developing the work performance of teachers, so that the goals and objectives of the school are more effectively achieved, while at the same time benefiting teachers in terms of recognition of performance, professional development and career guidance. Our work is useful for easy user interface. We are planning to utilize the powerful database management, data retrieval and data manipulation. Which will provide more ease for managing the data than manually maintaining in the documents. Our work is useful for saving valuable time and reduces the huge paperwork.

## List of figures

* ER-Diagram
* Schema Diagram

## List of tables

* Staff
* Teaching Staff
* Non-Teaching Staff
* Attendance
* School
* Department
* Courses
* Student
* Roles

## Introduction

* 1. **Motivation**

All the modules in college administration are interdependent. They are maintained manually. So, they need to be automated and centralized as, Information from one module will be needed by other modules.

* 1. **Objectives**

With that in mind, we overhauled the existing Teachers Database Management System and made necessary improvement to streamline the processes. Administrators using the system will find that the process of recording and retrieving teacher's information and managing their classes, lectures including marking of attendance, is now a breeze. In general, this project aims to enhance efficiency and at the same time maintain information accurateness. Later in this report, features and improvement that allow achievement to this goal will be demonstrated and highlighted.

## Problem Statement

* 1. **Scope**

TMS is intended to help the any institute that wants to store their staff and management records into the computer. Our software is specially designed for an educational institute. Our TMS software will fulfil all the requirement of an educational institute. It will store all information of the Staff. It will also store the information of the students, courses and departments of university. Management will get information of any student who studied/studying in this institute till now easily.

* 1. **Planning**

1. The teacher management system facilitates the user in following respects.
   1. Data Entry: Entering the correct and reliable data to the proposed system is the most important factor. The proposed system will facilitate the user with user-friendly screen, which will help the users to enter the correct data to proposed system easily. Different checks have been applied in the program for the validity of data so that wrong information cannot be entered into the system easily.
   2. Updating: Mistake in the entry process are possible to occur. These mistakes needed to be corrected at the right time so that the data remain correct. Any mistake during the entry process can be corrected through this option.
   3. Deletion: We can easily delete a particular record by requesting through entering the identity number or name of an item. The specified record is deleted from the database, which is not required anymore. This rarely occurs because most of the record are stored permanently, which may be referenced in later time.

## Tools & Technology Used

**Software Requirements**

1. HTML/CSS
2. JavaScript
3. Node-JS
4. MySQL

**Hardware Requirement**

1. Processor: - Pentium
2. RAM: - 512MB
3. Memory: 1GB

## Database Design

**Functional Requirements-**

### CREATION OF NEW RECORD:

This function creates a record for a new student

### DELETION OF RECORD:

This function is used to delete the existing record of any student.

### UPDATION IN RECORD:

This function updates the information in a record of any student.

### DISPLAY OF DATA IN RECORD:

This function displays the record of the students.

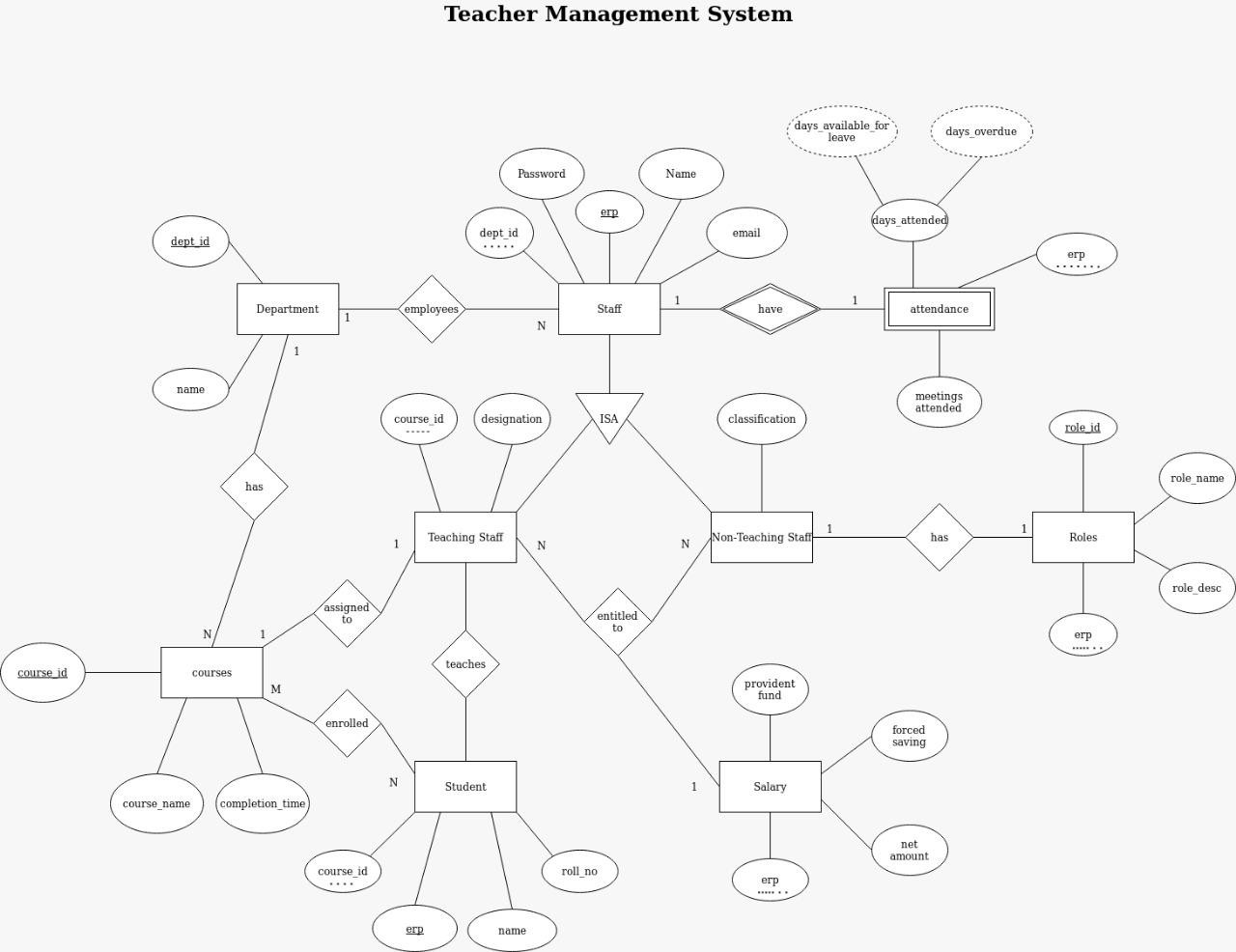
### SEARCHING A RECORD:

This function is used to search the database.

**Non- Functional Requirements-**

1. **Security:** Only authorized users can access the system with username and password.
2. **Performance:** Easy tracking of records and updating can be done.
3. **User Friendly**: The System is very interactive.
4. **Maintainability**: Backups for database are available.

**ER Diagram**



1. **Database Schema Schema Table**
   * **Staff**

|  |  |  |  |
| --- | --- | --- | --- |
| erp | name | Dept\_id | password |

* + **Attendance**

|  |  |  |  |
| --- | --- | --- | --- |
| Day\_attended | Day\_avail\_leave | Day\_overdue | Meetings\_attended |

* + **Teaching\_Staff**

|  |  |  |
| --- | --- | --- |
| erp | Course\_id | Designation |

* + **Non-Teaching\_Staff**

|  |  |
| --- | --- |
| erp | classification |

* + **roles**

|  |  |  |  |
| --- | --- | --- | --- |
| Role\_id | Role\_name | Role\_desc | erp |

* + **department**

|  |  |
| --- | --- |
| Dept\_id | Dept\_name |

* + **courses**

|  |  |  |
| --- | --- | --- |
| Course\_id | Course\_name | Completion\_time |

* + **student**

|  |  |  |
| --- | --- | --- |
| erp | name | rollno |

1. **DDL(Data Definition Language)**

DDL or Data Definition Language actually consists of the SQL commands that can be used to define the database schema. It simply deals with descriptions of the database schema and is used to create and modify the structure of database objects in the database.

**1 .Create-**

is used to create the database or its objects (like table, index, function, views, store procedure and triggers).

CREATE TABLE IF NOT EXISTS department(

dept\_id INT PRIMARY KEY,

name VARCHAR(20) NOT NULL

);

CREATE TABLE IF NOT EXISTS staff(

erp INT(12) PRIMARY KEY,

dept\_id INT NOT NULL,

name VARCHAR(20) NOT NULL,

password VARCHAR(20) NOT NULL,

email\_id VARCHAR(30) NOT NULL UNIQUE,

FOREIGN KEY (dept\_id) REFERENCES department(dept\_id) ON DELETE CASCADE ON UPDATE CASCADE

);

CREATE TABLE IF NOT EXISTS attendance(

attendance\_id INT PRIMARY KEY,

erp INT NOT NULL,

meetings\_attended INT NOT NULL DEFAULT 0,

days\_attended INT NOT NULL DEFAULT 0,

FOREIGN KEY (erp) REFERENCES staff(erp) ON DELETE CASCADE ON UPDATE CASCADE

);

CREATE TABLE IF NOT EXISTS salary(

erp INT PRIMARY KEY,

provident\_fund INT,

forced\_saving INT,

net\_amt DECIMAL(10,3) CHECK (net\_amt > 100)

);

CREATE TABLE IF NOT EXISTS courses(

course\_id INT PRIMARY KEY,

course\_name VARCHAR(20) NOT NULL,

completion\_time INT NOT NULL DEFAULT 4

);

CREATE TABLE IF NOT EXISTS teaching\_staff(

erp INT PRIMARY KEY,

course\_id INT NOT NULL,

designation VARCHAR(20) NOT NULL,

FOREIGN KEY (course\_id) REFERENCES courses(course\_id) ON DELETE CASCADE ON UPDATE CASCADE

);

CREATE TABLE IF NOT EXISTS non\_teaching\_staff(

erp INT NOT NULL,

classification VARCHAR(20) NOT NULL

);

CREATE TABLE IF NOT EXISTS roles(

role\_id INT PRIMARY KEY,

erp INT NOT NULL,

role\_name VARCHAR(20) NOT NULL,

FOREIGN KEY (erp) REFERENCES staff(erp) ON DELETE CASCADE ON UPDATE CASCADE

);

CREATE TABLE IF NOT EXISTS student(

stud\_erp INT PRIMARY KEY,

course\_id INT NOT NULL,

name VARCHAR(20) NOT NULL,

roll\_no INT NOT NULL UNIQUE,

FOREIGN KEY (course\_id) REFERENCES courses(course\_id) ON DELETE CASCADE ON UPDATE CASCADE

);

1. **DML(Data Manipulation Language)**

The SQL commands that deals with the manipulation of data present in the database belong to DML or Data Manipulation Language and this includes most of the SQL statements.

**2. Drop-** is used to delete objects from the database.

DROP DATABASE IF EXISTS db\_teacher;

CREATE DATABASE db\_teacher;

1.**INSERT**- is used to insert data into a table

INSERT INTO courses VALUES (1001,"Physics",4);

INSERT INTO courses VALUES (1002,"DBMS",4);

INSERT INTO courses VALUES (1003,"MIT",4);

INSERT INTO courses VALUES (1004,"DELD",4);

INSERT INTO courses VALUES (1005,"Humanity",4);

-- Teaching Staff

INSERT INTO teaching\_staff VALUES (13218001,1001,"Hitman Pro Max");

INSERT INTO teaching\_staff VALUES (13218003,1002,"Auror");

INSERT INTO teaching\_staff VALUES (13218005,1005,"Professional Retard");

INSERT INTO teaching\_staff VALUES (13218007,1004,"CKMKB");

-- Non Teaching Staff

INSERT INTO non\_teaching\_staff VALUES (13218002,"Hotel Managment");

INSERT INTO non\_teaching\_staff VALUES (13218004,"Care Taker");

INSERT INTO non\_teaching\_staff VALUES (13218006,"Coin Eater");

-- Roles

INSERT INTO roles VALUES (001,13218002,"Managment");

INSERT INTO roles VALUES (002,13218004,"Managment");

INSERT INTO roles VALUES (003,13218006,"Coin Eater");

**2.UPDATE-**is used to update existing data within a table.

CREATE TABLE IF NOT EXISTS student(

stud\_erp INT PRIMARY KEY,

course\_id INT NOT NULL,

name VARCHAR(20) NOT NULL,

roll\_no INT NOT NULL UNIQUE,

FOREIGN KEY (course\_id) REFERENCES courses(course\_id) ON DELETE CASCADE ON UPDATE CASCADE

);

**3.DELETE- DELETE-**is used to delete records from a database table.

CREATE TABLE IF NOT EXISTS student(

stud\_erp INT PRIMARY KEY,

course\_id INT NOT NULL,

name VARCHAR(20) NOT NULL,

roll\_no INT NOT NULL UNIQUE,

FOREIGN KEY (course\_id) REFERENCES courses(course\_id) ON DELETE CASCADE ON UPDATE CASCADE

);

1. **DCL-(Data Control Language)**

DCL includes commands such as GRANT and REVOKE which mainly deals with the rights, permissions and other controls of the database system.

1. **Triggers**-A trigger is a stored procedure in database which automatically invokes whenever a special event in the database occurs. For example, a trigger can be invoked when a row is inserted into a specified table or when certain table columns are being updated.
2. -- Triggers
3. -- TRIGGER FOR ATTENDANCE
4. DELIMITER $$
5. CREATE TRIGGER dns\_attendance
6. AFTER INSERT ON staff
7. FOR EACH ROW
8. BEGIN
9. IF NOT EXISTS (select \* from attendance where erp=new.erp) THEN
10. insert INTO attendance values (new.erp%100, new.erp, 0, 0);
11. END IF;
12. IF NOT EXISTS (select \* from salary where erp=new.erp) THEN
13. insert INTO salary values (new.erp, 0, 0,100);
14. END IF;
15. END$$
16. DELIMITER ;
17. **PLSQL Procedure/Function**-"A **procedures** or **function** is a group or set of SQL and PL/SQL statements that perform a specific task."  
    A function and  procedure is a named PL/SQL Block which is similar . The major difference between a procedure and a function is, a function must always return a value, but a procedure may or may not return a value.

DELIMITER $$

CREATE PROCEDURE inc\_sal()

BEGIN

DECLARE DONE INT DEFAULT 0;

DECLARE ERP INT;

DECLARE P\_FUND INT;

DECLARE F\_FUND INT;

DECLARE NET\_AMT DECIMAL(10,3);

DECLARE UPD\_AMT DOUBLE(10,3);

DECLARE CU1 CURSOR FOR SELECT \* FROM salary;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET DONE = 1;

OPEN CU1;

REPEAT

FETCH CU1 INTO ERP,P\_FUND,F\_FUND,NET\_AMT;

SET UPD\_AMT = NET\_AMT + NET\_AMT\*0.25;

UPDATE salary SET net\_amt = UPD\_AMT WHERE erp = ERP;

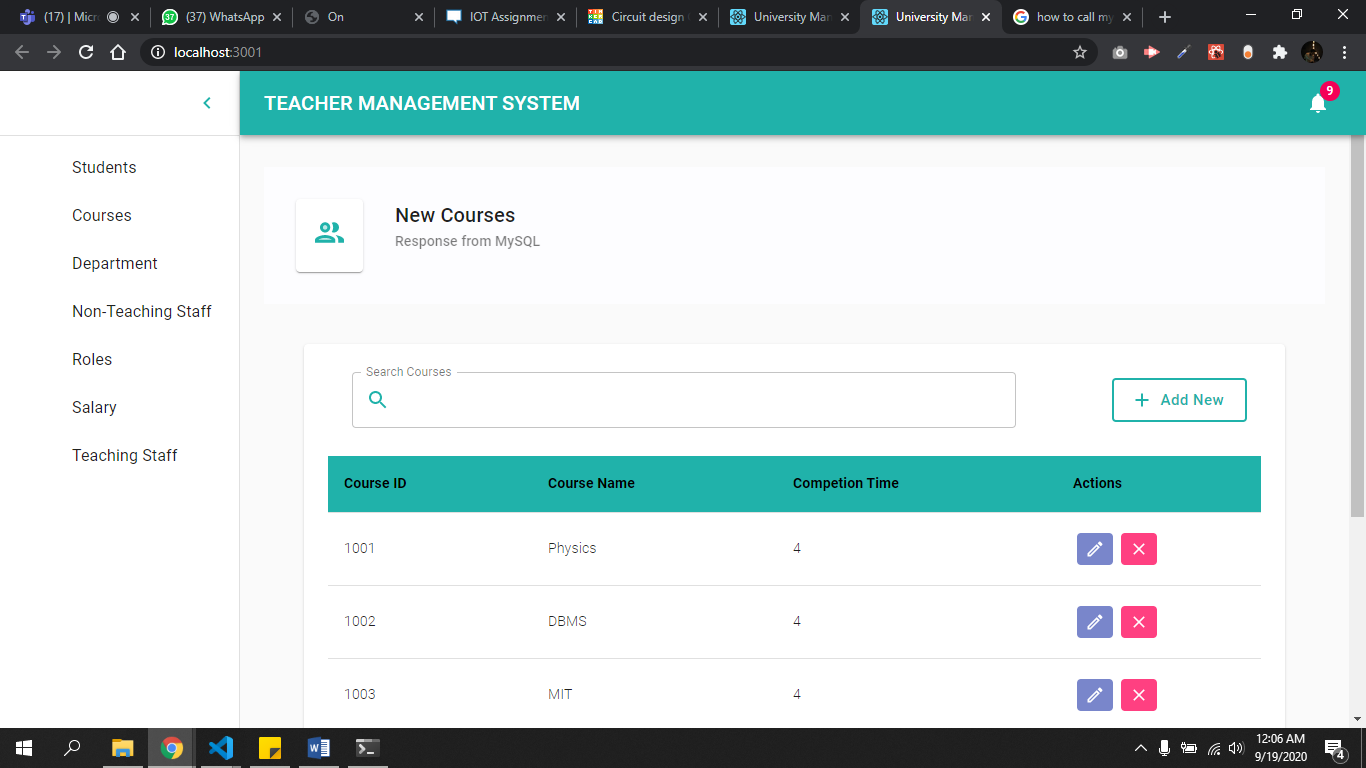
UNTIL DONE

END REPEAT;

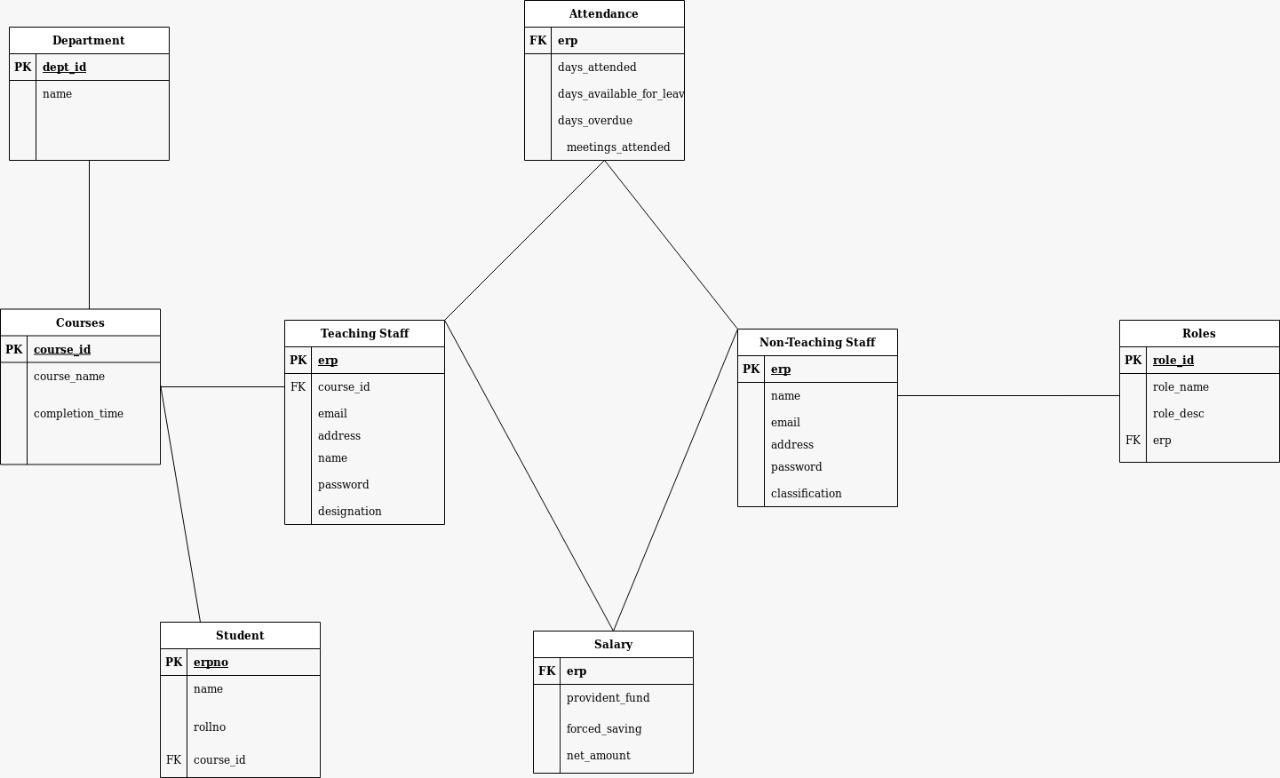
END $$

DELIMITER ;

1. Frontend GUI-



**Relational Database Design**



**Normalized Database Design**

* **1st Normal Form:**

Student can have multiple course\_id so First normalized form is needed and therefore a separate courses table was created with course\_id as primary key.

Due to inexistence of candidate keys

* **2nd Normal Form:**

Already in 2nd Normal Form

* **3rd Normal Form:**

Already in 3rd Normal Form

## Combined University Management ER Diagram

